第 25 卷第 3 期 半 导 体 学 报 Vol. 25, No. 3
2004 年 3 月 CHINESE JOURNAL OF SEMICONDUCTORS Mar., 2004

An Efficient Partitioning Method in Quadratic Placement*

Lü Yongqiang, Hong Xianlong, Hou Wenting, Wu Weimin and Cai Yici

(*Department of Computer Science and Technology, Tsinghua University, Beijing* 100084, *China*)

Abstract: A method of combining the MFFC clustering and hMETIS partitioning based quadratic placement algorithm is proposed. Experimental results show that it can gain good results but consume long running time. In order to cut down the running time, an improved MFFC clustering method (IMFFC) based Q-place algorithm is proposed. Comparing with the combining clustering and partitioning based method, it is much faster but with a little increase in total wire length.

Key words: partitioning; clustering; Q-place; MFFC; IMFFC; hMETIS
EEACC: 2570 **CCACC**: 7410D
CLC number: TN402 **Document code**: A **Article ID**: 0253-4177(2004)03-0272-07

1 Introduction

Quadratic placement (Q-place)[1] is a classic and efficient placement method based on quadratic programming optimization, adopting the quadratic wire length as the objective function. It first converts the placement problem into a quadratic programming model, and then uses proper means to solve the model. Q-place has been broadly used in nowadays placement tools for standard cell layout design. However, as the scale of ICs increases rapidly, the Q-place cannot solve such large-scale layout problems fast enough without any preprocessing. Hierarchical methodology is a good approach for reducing the size of problem and speeding up the running time. Some clustering or partitioning techniques can be used as a preprocessor in hierarchical method. There are a lot of clustering methods, random-walk based clustering[2], greedy clustering[3], MFFC clustering[4]; and partitioning methods such as the spectrum-based partitioning method[5], the generalization of the FM-algorithm with look-ahead scheme[6], and some two-way or multilevel partitioning methods. For the purpose of reducing the size of problem without lowering the solution quality, two issues must be taken into account in consideration of clustering or partitioning methods. First, the inner nets in clusters or partitions must be as many as possible. Second, the interconnected nets among clusters or partitions must be as few as possible. As a successful application of clustering scheme in Q-place, the greedy cluster-based Q-place algorithm is proposed in Ref. [3]. It gains excellent results but still has two shortcomings. First, the inner nets are not grouped as many as possible because it ignores the signal flow and logic dependency of the circuits. Second, the method is not considered reducing interconnected nets among clusters. In this paper the model of combining MFFC clustering and hMETIS partitioning method is selected, not only to group cells

* Project supported by National Natural Science Foundation of China (Nos. 60121120706, 60076016) and National Natural Science Foundation of USA (No. CCR-0096383), National Foundation Research (973) Program of China (No. G1998030403), and National High Technology Research and Development Program of China (No. 2002AA1Z1460)

Lü Yongqiang male, was born in 1980, PhD candidate, majoring VLSI layout design.

Received 5 June 2003, revised manuscript received 26 August 2003

in a cluster with connection as close as possible, but also to reduce the interconnected nets among clusters. The MFFC clustering can create little clusters with high quality, and the hMETIS can partition the MFFC-clustered circuits with less interconnected nets (cutsize) among the partitions. The hMETIS is an available partitioning package; it also can take the standard deviation into account. However, the hMETIS leads to a long running time except for the low cutsize. For the efficiency of the model, an improved MFFC clustering algorithm is developed to achieve a fast running time with nearly the same total wire length. Experimental results show that the algorithm can cut down the running time of placement remarkably especially with large-scale circuits.

The hierarchical placement process used in this paper can reduce the placement time and improve the placement quality. Particularly in very-large-scale standard cell placement, it acts as the main placement means instead of the flatten Q-place or other flatten ways.

2 Q-place

The quadratic placement is a determinable method based on quadratic programming optimization. Its objective function is the quadratic wire length. The Q-place begins with modeling the problem into a quadratic problem. Then it solves the problem by certain means and obtains the global placement solution.

In the Q-place used in this paper, the problem is modeled into a linear constraint quadratic problem (LQP)[1], and it is solved to obtain a placement solution in each partition level. The partition level is a state of each level of the partition tree, partitioning a parent region into two son regions in each partition step. The number of regions in ith partition level is 2^i.

First of all, a circuit is presented as a hypergraph $G(V, E)$, $V = V_1 \cup P$, $V_1 = \{v_1, v_2, \cdots, v_n\}$, representing all movable cells in the circuit; $P = \{p_1, p_2, \cdots, p_q\}$, representing all fixed cells in the circuit such as primary outputs and primary inputs. The hyperedge set $E = \{e_1, e_2, \cdots, e_m\}$, representing the connection of the circuit. Every edge is assigned to a weight $w(e)$. The coordinate of cell i is (x_i, y_i).

The objective function of the global placement is the weighted sum of the squared lengths of the nets[1]; form in matrix is

$$\Phi(x, y) = x^T \boldsymbol{C} x + \boldsymbol{d}_x^T x + y^T \boldsymbol{C} y + \boldsymbol{d}_y^T y \quad (1)$$

x and y are the vectors of the cells' coordinates, and they are just to be solved. The $n \times n$ connection matrix $\boldsymbol{C} = (c_{ij})$ for G represents the weight information of edges. For each edge $e_k = (v_i, v_j)$, $c_{ij} = c_{ji} = -w(e_k)$; the diagonal entry c_{ii} is equal to or larger than the sum $\sum_{j=1, j\neq i}^{n} c_{ij}$, all other entries of $\boldsymbol{C}$ are zeros. The vector $\boldsymbol{d}_x$ and $\boldsymbol{d}_y$ are generated by the connection between the cells and pads. In order to distribute the cells evenly on the chip, distribution constraints are added:

$$\sum_{i \in S_r} x_i / |S_r| = u_r, \sum_{i \in S_r} y_i / |S_r| = v_r \quad (2)$$

S_r is the cell set in the region r; $(u_r\ v_r)$ is the center of the region r; and the $|S_r|$ is the number of the cells in S_r.

So, the LQP obtained is

$$\min\{\Phi(x, y) = 1/2 x^T \boldsymbol{C} x + \boldsymbol{d}_x^T x + 1/2 y^T \boldsymbol{C} y + \boldsymbol{d}_y^T y \mid \boldsymbol{A}^{(m)} x = \boldsymbol{u}^{(m)}; \boldsymbol{A}^{(m)} y = \boldsymbol{v}^{(m)}\} \quad (3)$$

In mth placement-partition level, matrix $\boldsymbol{A}$ is a $2^m \times n$ matrix. In matrix $\boldsymbol{A} = (a_{ij})\ 2^m \times n$, the a_{ij} equals to $1/|S_j|$ if cell i belongs to region j or a_{ij} is zero. And the number of total none-zero entries is equal to n.

The Lagrange multipliers method is employed to solve the LQP[1]. The corresponding Lagrange function is

$$\min\{\Phi(x) = 1/2 x^T \boldsymbol{C} x + \boldsymbol{b}_x^T x - \lambda(\boldsymbol{A}^{(m)} x - \boldsymbol{u}^{(m)})\} \quad (4)$$

where λ is the Lagrange multiplier. Making the gradient equal to zero, the linear system will be obtained:

$$\begin{bmatrix} \boldsymbol{C} & -\boldsymbol{A}^{(m)} \\ -\boldsymbol{A}^{(m)T} & 0 \end{bmatrix} \begin{bmatrix} x \\ \lambda \end{bmatrix} = - \begin{bmatrix} \boldsymbol{b}_x \\ \mu^{(m)} \end{bmatrix} \quad (5)$$

To solve this equation, the cells optimal location in x-direction can be obtained, and the y-direction optimal solution can be obtained in the same way. The LQP is solved repeatedly until reaching the desired placement-partition level. After accomplishing solving the LQP at the desired placement-partition level, the global optimal placement of the cells is acquired.

To solve this problem, equation (5) needs to be solved, and the time complexity of solving the equation would amount to at least $O(n\lg n)$. So, to decrease the scale of the quadratic programming problem is very important to solve the large-scale standard cell placement.

3 Combining clustering and partitioning

In order to decrease the problem scale in very large-scale standard cell placement, the method of combining the MFFC clustering and hMETIS partitioning is introduced to simplify the circuits in this paper. The Q-place runs on the clustered circuits to give the global placement of cells.

3.1 MFFC clustering

The MFFC clustering technique is composed of the MFFC decomposition technique and the MFFC splitting technique.

3.1.1 MFFC decomposition technique

The MFFC decomposition technique was first proposed for combinational circuits in Ref. [8], and was frequently used in later research. For clarity, to define

(1) Output(v) as the set of nodes which are the fanouts of node v, that is, the node w in output(v) must be the terminal of one of the output nets of the node v.

(2) C_v as the subgraph of the logic gates (excluding primary inputs (PIs)) consisting of v and its predecessors such that any path connecting a node in C_v and v lies entirely in C_v.

(3) FFC_v as the fanout-free cone (FFC) at node v, it is a cone of v such that for any node w (not v) in FFC_v, output(w) is included by FFC_v.

(4) $MFFC_v$ as the FFC of v such that for any non-PI node w, if ouput(w) is included by $MFFC_v$, then w is in $MFFC_v$. The node v is called as the root of the $MFFC_v$.

In general, the gates in a single $MFFC_v$ can be considered closely related because the MFFC clustering technique guarantees it.

The MFFC decomposition technique is to cluster the cells into MFFC sets from the corresponding POs, and POs are the cells with which decompositions begin, initially the primary outputs. The decomposition technique executes as Fig. 1 illustrates.

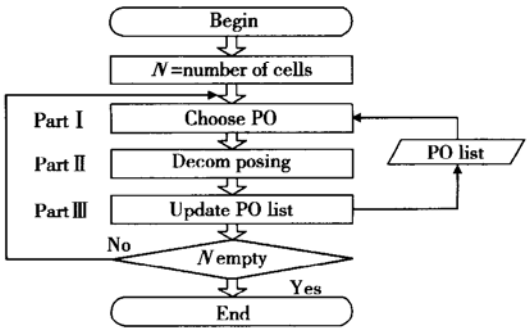

Fig. 1 MFFC decomposition flow

At the beginning of the flow, let N be the set of the cells of the circuit; let PO list be the list of primary outputs in arbitrary sequence initially. In the part I, the PO v selected is to be the cell that decomposition begins with, and is put into the corresponding $MFFC_v$ first. In the part Ⅱ, trace cells from each input net of v, put the cell w traced into the $MFFC_v$ as soon as output(w) is included in the $MFFC_v$, and let $N = N - MFFC_v$. In the part Ⅲ, the new trace-from cells are updated into the PO list; those do not need the conditions of clustering into current MFFC. For example, a circuit as Fig. 2 shows will be decomposed into A, B, C, D, E five clusters. The cluster A is decomposed from the set of only cell V (one initial PO), and it absorbs cells V1 because of the only output(V1), the cell V, is in the set. While the cell s cannot be clustered into

the set because there is one output(s), the cell V2, not in the set.

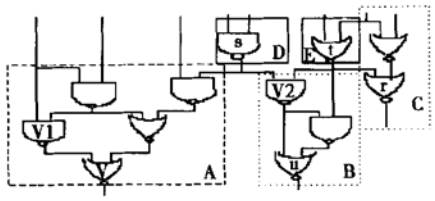

Fig. 2　A MFFC decomposition example

3.1.2　MFFC splitting technique

After MFFC decomposition, a MFFC splitting process will be adopted to split the large MFFC sets into smaller ones so that the area of the clusters could not be too large.

A splitting tree (ST) is employed to complete the MFFC splitting process. The nodes in the ST stand for MFFCs; one node is one MFFC. If node r is $MFFC_v$, sons of r are those decomposed from one of the cells in the input(v) (the fanins of v). Let the root of the ST be the MFFC to be divided, and keep creating the sons of the nodes in the ST recursively until the area of the MFFC to be split is proper. For example, the splitting tree as the Fig. 3 shows is to be split as {u1}, {u2}, {u3, u4}, {u7}, {u5, u6, u8}.

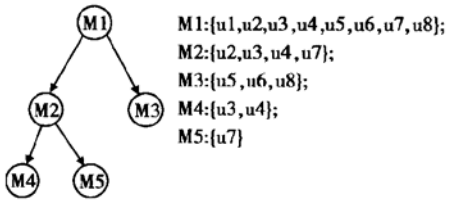

Fig. 3　A splitting tree and its splitting result

The decomposition technique used in constructing ST is the same to the decomposition technique described above. Every time splitting a big MFFC, a one-cell MFFC will appear.

3.2　hMETIS package

The hMETIS[9] package is selected to partition the hypergraph of the clustered-circuits after MFFC clustering is employed. HMETIS includes three phases: coarsening phase, initial partitioning, and refinement phase. In coarsening phase, vertexes of the hyper-graph inputted are grouped to construct a smaller hypergraph. Then in initial partitioning phase, mainly the Fiduccia-Mattheyses (FM) algorithm is employed. At last phase, the hypergraph is returned to original graph and refined based on FM algorithm.

The proper partition number is determined by experiment according to the corresponding circuits, constrained by proper problem scale for Q-place. By default the parameter is set to one tenth of the total cell number.

The hypergraph of the circuit which will be partitioned by hMETIS is constructed by making the MFFC clusters be the nodes with weights of area and making the nets be the hyperedges with weights of cluster net weights. According to the weight of the nodes, the evenness of the partition obtained can be controled.

3.3　Combining the two

In the large-scale standard cell placement, a preprocessor should be added before the Q-place, in which the large circuits are clustered or partitioned into small circuits. So the Q-place works on the clustered or partitioned circuits. Two ways, clustering and partitioning, are often used to preprocess circuits. However, clustering can get high-quality clusters but is not easy to reduce the interconnect nets among clusters; and partitioning can reduce interconnect nets but is not good at reducing the inner nets of partitions. So an idea to combine the two together is advocated. The preprocessor used in this paper, the combining of MFFC clustering and hMETIS can reduce the inner connection in clusters and interconnected nets among partitions remarkably. From the experimental results in Section 5, the good total wire length proves it. So, the entire hierarchical placement process can be described as Fig. 4 illustrates.

The part of clustering executes the MFFC clustering, collecting the cells with closest connections. The partitioning part regroups the MFFC

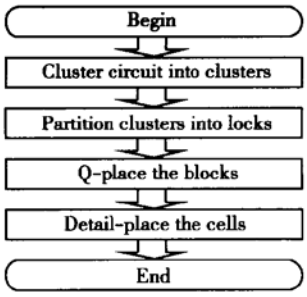

Fig. 4 Hierarchical placement process

clusters into partitions. Then the Q-place processes the clusters to obtain the global cell placement solution.

4 Improved MFFC clustering

Unsatisfied with the running time of combining method, the improved MFFC clustering is introduced to preprocess circuits. The MFFC clustering method can create high quality clusters rapidly, but the number of clusters may be too large to do the global placement fast. If the MFFC clustering method can create a proper number of clusters, it can make the placement of large scale run faster. It is found that the process of the MFFC decomposition may be too strict because only when the cell v meets the condition that output(v) is included in the current MFFC set, can it be included. In this algorithm, cell v will be included into the MFFC when the output(v) is mainly included in it, and it is proved that it can help creating proper clusters with even area. The IMFFC-based algorithm is called as merging algorithm.

First, let us see these properties of MFFC clustering:

(1) The output nets of a cluster must begin with the root cell of the cluster. This is obvious because of the decomposition technique.

(2) The input nets of a cluster must begin with the cells that are generated as new POs.

The proof in details is omitted here. The connection information of the MFFCs conveniently can be generated when cells are clustered. According to the connection information recorded in the sets, a merging process is appended after the primary MFFC clustering. The merging algorithm used is described as following.

```
Algorithm merging
  while (MergeControl> 0){
    for each MFFC set
      MergingProperSet();
    end for
    MergeControl= MergeControl- controlStep;
  end while
End Algorithm
Process MergingProperSet
  for each set connecting with the current
      select the set with maximum connection degree to merge;
      update the connection information;
  end for
End Process
```

The MergeControl and controlStep used in the merging algorithm are experiential parameters. The MergeControl is chosen as the average number of the cells in clusters and the controlStep is chosen as 1 in our development. They are employed only to control the merging cycling numbers, i. e. the average set-size. Because larger the MergeControl is and less the controlStep is, more the while cycling number is, so the larger set-size will be obtained. They are determined in experiments. The connection degree of the set i and j is counted as

$$D_{ij} = \sigma C_{in}/C_{out} \quad (6)$$

C_{in} is the interconnection of the two sets; C_{out} is the outer connection of the two sets; σ is the area control factor, determined by experience, and taken as $1/S$, S is the total area of the two sets. Merging the proper set is to select the set with maximum connection degree calculated with Eq. (6), and updating connection is to delete the merged connections.

5 Experimental results

The combining and the improved MFFC clustering method have been implemented on Sun

workstation V880 in C language. The package of hMETIS V1.5.3 is obtained on WWW at http://www.cs.umn.edu/~metis. Six LEF/DEF benchmarks obtained from real circuits in industrial fields were used. All experimental results were obtained after the global placement completes and before the detail placement begins, and the number of clusters is the same to what the greedy cluster-based Q-place algorithm used. Table 1 shows the characteristics of these benchmarks.

Table 1 Characteristics of the circuits

Case	# cell	# net
U05614	32112	36452
U08421	48168	54741
CPU	48876	49204
U11228	64224	72966
U14035	80280	91127
U28070	160560	181932

First of all, let us see the experiment with only partitioning. Because of the too much CPU time consumed, only the data of case CPU is presented: wire length: 5445162, CPU time: 1265s. It is almost the same short wire length as Table 2 but much more time is consumed. Also, the flatten Q-place would consume even more time on those cases. That is why the model of hierarchical placement process with combining of clustering and partitioning is adopted.

Then, let us see the comparison between our combining Q-place and the greedy cluster-based Q-place[3] as Table 2 illustrates. The comparisons can be presented in three aspects: total wire length, CPU time (running time), and standard deviation.

(1) Total wire length. The improvement in total wire length can be seen in Table 2. It can be seen that the method of combining MFFC clustering and hMETIS (the column of comb in the table) can decrease the total wire length for 28% in average. On the circuit of CPU, it can even decrease the wire length for 39%.

(2) Running time. From Table 2, It can be seen that the running time our method used may be a little long. The running time is mostly taken in running partitioning (about 1/4).

(3) Standard deviation. Our cluster area is very even and most clusters have a nearly average area; while that of greedy cluster-based method ranges much. For example, with circuits of CPU, our results range from 10 to 10^2 in order, while the greedy method ranges from 10 to 10^3.

Table 2 Comparison between the combining and Ref. [3]

Case	Total wire length			CPU time/s	
	comb	greedy	- %	comb	Gree
U0561	7612377	1034440	26	459.7	317.2
U0842	1290552	1642805	21	700	491
CPU	5377955	8811934	39	909.5	492.2
U1122	1698727	2284372	26	926.8	650.4
U1403	2137201	2958767	28	1027	738.5
U2807	4512022	6231557	28	1333	999.4

Table 3 gives the comparison between the IMFFC and the combining:

(1) Total wire length. Because hMETIS has a random-number driven procedure in its initial partition phase, it is not stable enough in all cased. In some cases, such as U08421 and U14035, IMFFC can outperform hMETIS. The negative values in the colomn(+ %) of Table 3 show that the IMFFC obtains less total wire length than hMETIS in those cases. The IMFFC can only increase 4% of wire length in average in comparison with hMETIS.

(2) Running time. The IMFFC can cut down the running time remarkably. In large circuits, it can even cut down about 50% of the running time such as U14035 and U28070.

(3) Area evenness. The area is also even, but not as good as the method of combining. For some circuits, its result is not even such as U08421.

Table 3 Comparison between IMFFC and combining

Case	Total wire length			CPU time/s	
	IMFFC	comb	+ %	IMFFC	comb
U05614	8267863	7612377	9	304.4	459.7
U08421	12508793	12905521	- 3	514.5	700
CPU	5626875	5377955	4	656.4	909.5
U11228	18621614	16987275	9	371.3	926.8
U14035	20975662	21372019	- 2	527.1	1027
U28070	48429853	45120227	7	567.1	1333

6 Conclusion

The method of combining clustering and partitioning can improve the quality of placement remarkably, and the improved MFFC clustering method (IMFFC) based Q-place algorithm can gain excellent results with less running time in comparison with the combining method. In future, congestion and other kinds of placement for example mixed-mode placement with big macros will be taken into account.

References

[1] Hou Wenting, Yu Hong, Hong Xianlong, et al. A new congestion-driven placement algorithm based on cell inflation. Chinese Journal of Semiconductors, 2001, 22(3): 275

[2] Hagen L, Kahng A B. A new approach to effective circuit clustering. Int'l Conf Computer-Aided Design, 1992: 422

[3] Yu Hong, Hong Xianlong, Wu Weimin, et al. CASH: a novel quadratic placement algorithm for very large standard cell layout design based on clustering. Proceedings of the 5th International Conference on Solid-State and Integrated Circuit Technology, 1998: 496

[4] Cong Jason, Xu Dongmin. Exploiting signal flow and logic dependency in standard cell placement. Proceedings of the conference on Asia Pacific design automation Conference, 1995: 399

[5] Chan P, Schlag M, Zien J. Spectral K-way ratio-cut partitioning and clustering. Proc 30th ACM/IEEE Design Automation Conf, 1993: 123

[6] Sanchis L. Multiple-way network partitioning. IEEE Trans Comput, 1989, 38: 62

[7] Cong J, Smith M. A bottom-up clustering algorithm with applications to circuit partitioning in VLSI designs. Proc ACM/IEEE Trans VLSI Syst, 1994, 2: 137

[8] Cong J, Ding Y. On area/depth trade-off in LUT-based FPGA technology mapping. Proc 30th ACM/IEEE Design Automation Conf, 1993: 213

[9] Karypis G, Kumar V. hMETIS, a hypergraph partitioning package version 1.5.3, 1998

一种应用于二次布局的有效划分方法*

吕勇强 洪先龙 侯文婷 吴为民 蔡懿慈

(清华大学计算机科学技术系, 北京 100084)

摘要: 提出了一种基于二次布局的结合 MFFC 结群和 hMETIS 划分的算法. 实验表明: 这种方法能得到很好的布局结果, 但是运行消耗的时间比较长. 为了缩短划分在二次布局中运行的时间, 提出了一种改进的结群算法 IMFFC, 用它在二次布局中做划分. 与前者相比较, 这种方法虽然布局质量稍差, 但速度更快.

关键词: 划分; 结群; 二次布局; MFFC; IMFFC; hMETIS

EEACC: 2570 **CCACC**: 7410D

中图分类号: TN402 **文献标识码**: A **文章编号**: 0253-4177(2004)03-0272-07

* 国家自然科学基金(批准号: 60121120706 和 60076016), 美国国家自然科学基金(批准号: CCR-0096383), 国家高技术研究发展计划(批准号: 2002AA1Z1460) 和国家重点基础发展规划(批准号: G1998030403) 资助项目

吕勇强 男, 1980 年出生, 博士研究生, 专业方向为大规模集成电路布图布局.

2003-06-05 收到, 2003-08-26 定稿